PowerDNA DIO-403 User Manual

48-channel Digital I/O layer
for the PowerDNA Cube

September 2008 Edition
PN Man-DNA-DIO-403-0908

Version 3.3

© Copyright 1998-2008 United Electronic Industries, Inc. All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form by any means, electronic, mechanical, by photocopying, recording, or otherwise without prior written permission.

Information furnished in this manual is believed to be accurate and reliable. However, no responsibility is assumed for its use, or for any infringements of patents or other rights of third parties that may result from its use.

All product names listed are trademarks or trade names of their respective companies.

See UEI's website for complete terms and conditions of sale:
http://www.ueidaq.com/company/terms.aspx

Contacting United Electronics Industries

Mailing Address:
27 Renmar Avenue
Walpole, MA 02081
U.S.A.

For a list of our distributors and partners in the US and around the world, please see http://www.ueidaq.com/partners/

Support:
Telephone: (508) 921-4600
Fax: (508) 668-2350
Also see the FAQs and online "Live Help" feature on our web site.

Internet Support:
Support support@ueidaq.com
Web-Site www.ueidaq.com
FTP Site ftp://ftp.ueidaq.com

Table of Contents

Introduction ..... *iii*
Organization of this manual iii
Conventions iv
1 The DIO-403 Layer *1*
1.1 Device architecture 2
1.2 Layer capabilities 2
1.3 Layer connectors and wiring 3
2 Programming using the UeiDaq Framework *5*
2.1 Creating a session 5
2.2 Configuring the resource string 5
2.3 Configuring the timing 5
2.4 Reading data 6
2.5 Cleaning-up the session 6
3 Programming using the Low-Level API *7*
3.1 Data representation 7
3.2 Configuration settings 7
3.3 Channel list settings 8
3.4 Layer-specific commands and parameters 8
3.5 Using layer in ACB mode 8
3.6 Using layer in DMap mode 9
4 Appendix *11*
5 Glossary *13*
6 Index *17*

Introduction

This document outlines the feature-set and use of the DIO-403 layer.
This layer is a digital input and output module for the PowerDNA I/O Cube.

Organization of this manual

This PowerDNA DIO-403 User Manual is organized as follows:

Introduction
This chapter provides an overview of PowerDNA Digital Input Series board features, the various models available and what you need to get started.

The DIO-403 layer
This chapter provides an overview of the device architecture, connectivity, and logic of the DIO-403 layer.

Programming using the UeiDaq Framework High-Level API
This chapter provides an overview of the how to create a session, configure the session for digital data acquisition/output, and format relevant output.

Programming using the Low-Level API
Low-level API commands for configuring and using the DIO-403 series layer.

Appendix A - Accessories
This appendix provides a list of accessories available for DIO-403 series layer.

Appendix B - Layer Verification
This appendix outlines how to verify calibration for the DIO-403 series layer.

Glossary
This is an alphabetical listing of key terms you will encounter in working with the PowerDNA cube and test systems in general.

Index
This is an alphabetical listing of the topics covered in this manual.

Conventions

To help you get the most out of this manual and our products, please note that we use the following conventions:

Tips are designed to highlight quick ways to get the job done, or reveal good ideas you might not discover on your own.

Note Notes alert you to important information.

CAUTION! *Caution advises you of precautions to take to avoid injury, data loss, and damage to your boards or a system crash.*

Text formatted in **bold** typeface generally represents text that should be entered verbatim. For instance, it can represent a command, as in the following example: "You can instruct users how to run setup using a command such as **setup.exe**."

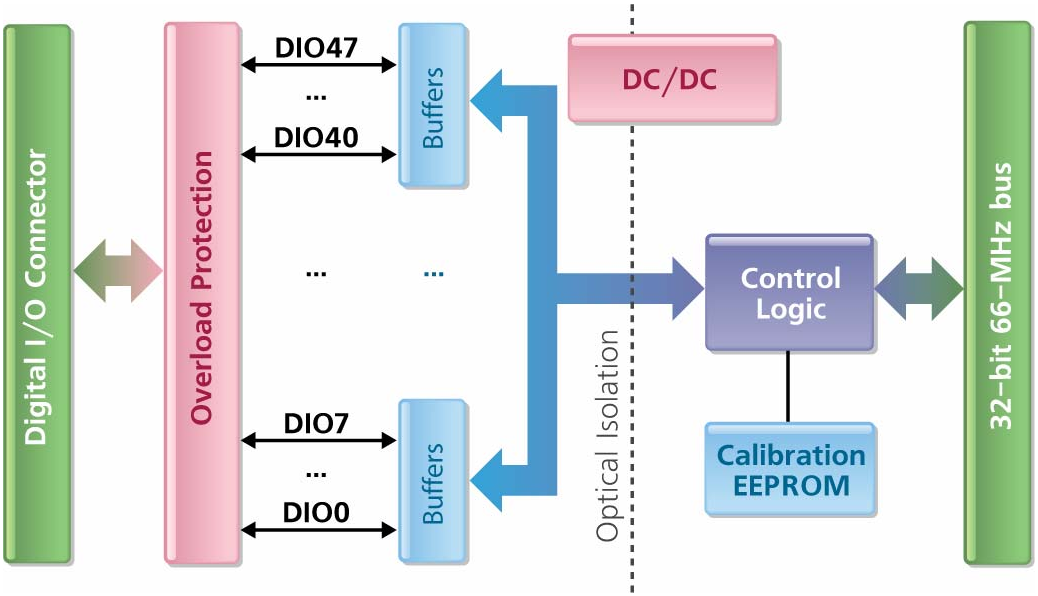
1 The DIO-403 Layer

The DIO-403 is a 48-line digital I/O layer that operates at TTL levels. Features:

- Triggering available on digital inputs
- Change-of-state detection on inputs
- Onboard FIFO memory: 128 32-bit words input / 128 32-bit words output
- Pattern output, I/O throughput rate is 10k samples/sec (20k aggregate)
- Digital Lines: 48 (direction selectable in groups of 8)
- Output Drive Capacity: 16 mA per pin (i.e., 16mA per channel)
- Input High Voltage: 2.4V
- Input Low Voltage: 0.8V
- Output High Voltage: 4.5V @ 2mA, 3V @ 16mA
- Output Low Voltage: 0.5V
- 22Ω current limiting resistors
- Timestamp resolution 15 ns
- Lines protected to +/-30V overvoltage peak-to-peak; 7kV ESD.
- Output protection: 140mA PTC fuse
- Isolation $350V_{rms}$
- Power Consumption: 1.2W + 0.007W/mA of load current

1.1 Device architecture

Architecturally, the DIO-403 is divided into a non-isolated logic part and an isolated part with I/O buffers.
Every line is protected from overvoltage and electrostatic discharge by a Harris SP720 overvoltage & ESD electronic protection array. One 22Ω current limiting resistor limits current in every line.



I/O lines are neither pulled-up nor pulled-down. Thus, connected equipment should actively drive these lines high or low. An unconnected or not driven input line will remain in an arbitrary state.
The layer employs 16,543 buffers with two-by-eight architecture. Thus, the user can select lines to be input or output with the granularity of eight lines at a time.

1.2 Layer capabilities

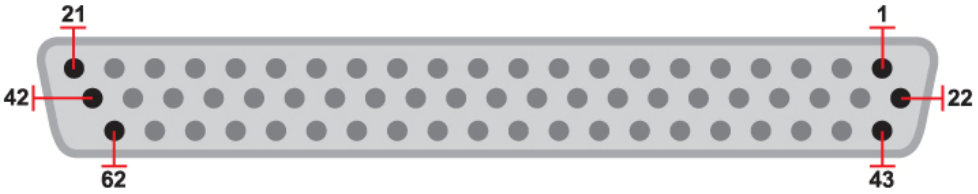
The DIO-403 layer is capable of single read/write into the registers as well as continuous clock reads and writes.
Current firmware supports single read/writes only.

1.3 Layer connectors and wiring

Wiring of DIO-403 is very simple. The user should wire input and output lines relative to *DGND*.

DB-62 (female)
62-pin connector:

62	N/C	42	DGND	21	+5V@140mA
61	DIO2	41	DIO1	20	DIO0
60	DIO5	40	DIO4	19	DIO3
59	DIO8	39	DIO7	18	DIO6
58	DIO11	38	DIO10	17	DIO9
57	DIO14	37	DIO13	16	DIO12
56	DIO17	36	DIO16	15	DIO15
55	DIO20	35	DIO19	14	DIO18
54	DIO23	34	DIO22	13	DIO21
53	DIO26	33	DIO25	12	DIO24
52	DIO29	32	DIO28	11	DIO27
51	DIO32	31	DIO31	10	DIO30
50	DIO35	30	DIO34	9	DIO33
49	DIO38	29	DIO37	8	DIO36
48	DIO41	28	DIO40	7	DIO39
47	DIO44	27	DIO43	6	DIO42
46	DIO47	26	DIO46	5	DIO45
45	TRIG1	25	TRIG2	4	TRIG0
44	DGND	24	N/C	3	N/C
43	DGND	23	DGND	2	DGND
		22	DGND	1	DGND



- *TRIGx* line is an input for external trigger.

2 Programming using the UeiDaq Framework

This section describes how to program the PowerDNA DIO-403 using the UeiDaq's framework API.
The UeiDaq framework is object oriented and its objects can be manipulated in the same manner from different development environments such as Visual C++, Visual Basic or LabVIEW.
The following section will focus on the C++ API but the concept stay the same no matter what programming language you use.
Please refer to the "UeiDaq Framework User Manual" to get more information on using other programming languages.

2.1 Creating a session

The Session object controls all operations on your PowerDNA device. Therefore, the first task is to create a session object:

```
CUeiSession session;
```

2.2 Configuring the resource string

The framework uses resource strings to select which device, subsystem and channels to use within a session. The resource string syntax is similar to a web URL:

```
<device class>://<IP address>/<Device Id>/<Subsystem><Channel
list>
```

For PowerDNA, the device class is **pdna**.
For example, the following resource string selects digital input channels 0,1,2,3 on device 1 at IP address 192.168.100.2: "pdna://192.168.100.2/Dev1/Di0:3"

```
session.CreateDIChannel("pdna://192.168.100.2/Dev1/Di0:3");
```

2.3 Configuring the timing

You can configure the DIO-403 to run in simple mode (point by point) or buffered mode (ACB mode).
In simple mode, the delay between samples is determined by software on the host computer.
In buffered mode, the delay between samples is determined by the DIO-403 on-board clock.

The following sample shows how to configure the simple mode. Please refer to the "UeiDaq Framework User's Manual" to learn how to use the other timing modes.

```
session.ConfigureTimingForSimpleIO();
```

2.4 Reading data

Reading data from the DIO-403 is done by using a reader object.
The following sample code shows how to create a scaled reader object and read samples.

```
// Create a reader and link it to the session's stream
CUeiDigitalReader reader(di_session.GetDataStream());

// read one scan, the buffer must be big enough to contain
// one value per channel
uInt16 data;
reader.ReadSingleScan(&data);
```

Writing data is done by using a writer object. The following sample shows how to create a writer object and write data .

```
// Create a writer and link it to the session's stream
CUeiDigitalWriter writer(do_session.GetDataStream());

// write one scan, the buffer must contain
// one value per channel
uInt16 data = 0xFEFE;
writer.WriteSingleScan(&data);
```

2.5 Cleaning-up the session

The session object will clean itself up when it goes out of scope or when it is destroyed. However, you can manually clean up the session (to reuse the object with a different set of channels or parameters) as follows.

```
session.CleanUp();
```

3 Programming using the Low-Level API

This section describes how to program the PowerDNA cube using the low-level API. The low-level API offers direct access to PowerDNA DAQBios protocol and also allows you to access device registers directly.
We recommend that you use the UeiDaq framework (*see Section 2 above*), which is easier to use.
You should need to use the low-level API only if you are using an operating system other than Windows.

3.1 Data representation

Internally, the DIO-403 presents the state of its line in two 32-bit words; each one contains 24 bits of data.

At the user level, read or write data is presented in an array of six bytes. Each byte represents one of the six I/O ports available.

3.2 Configuration settings

Configuration settings are passed in the `DqCmdSetCfg()` function.
Not all configuration bits apply to DIO-403 layers.

Following bits are used:

```
#define DQ_LN_MAPPED     (1L<<15)  // For WRRD (DMAP) devices (automatically selected)
#define DQ_LN_ACTIVE     (1L<<1)   // "STS" LED status
#define DQ_LN_ENABLED    (1L<<0)   // enable operations
```

`DQ_LN_ACTIVE` is needed to switch on "STS" LED on the CPU layer.
`DQ_LN_ENABLE` enables all operations with the layer

Layer-specific bits are the follows:

```
// enable ports 0..5 for write (otherwise they are in tristate - read)
#define DQ_DIO403_ENPORT5   (1UL << 13)
#define DQ_DIO403_ENPORT4   (1UL << 12)
#define DQ_DIO403_ENPORT3   (1UL << 11)
#define DQ_DIO403_ENPORT2   (1UL << 10)
#define DQ_DIO403_ENPORT1   (1UL << 9)
#define DQ_DIO403_ENPORT0   (1UL << 8)
```

Adding these bits into configuration word causes layer to switch respective ports into outputs.

3.3 Channel list settings

Channel list is not currently supported for this layer.

3.4 Layer-specific commands and parameters

There are three layer-specific function used to access and control the state of I/O lines:

- `DqAdv403SetIo()`
  This function selects which ports are inputs and which are outputs. Ports are represented as individual bits (port 0 – bit 0, etc.) while "1" represents output. Thus, to configure ports 0 thru 3 as inputs and ports 4 and 5 as outputs Mask = 110000b = 0x30

- `DqAdv403Write()`
  This function writes `data` to DIO-403 ports. Firmware writes data into ports regardless of the state of the port (input or output). User can write set the state of the output line by writing into ports and then enable output on all ports simultaneously.

- `DqAdv403Read()`
  This function returns the input status of all DIO-403 ports.

3.5 Using layer in ACB mode

The DIO-403 layer does not currently support Advanced Circular Buffer mode.

3.6 Using layer in DMap mode

```
#include "PDNA.h"
```

1. Start DQE engine

```
#ifndef _WIN32
    DqInitDAQLib();
#endif

    // Start engine
    DqStartDQEngine(1000*10, &pDqe, NULL);

    // open communication with IOM
    DqOpenIOM(IOM_IPADDR0, DQ_UDP_DAQ_PORT, TIMEOUT_DELAY,
&DQRdCfg);

    // Set hysteresis at this point
    DqAdv40xSetHyst(hd0, DEVNIN, 0x132, 0x2CA);

    // Receive IOM crucial identification data
    DqCmdEcho(hd0, DQRdCfg);

    for (i = 0; i < DQ_MAXDEVN; i++) {
        if (DQRdCfg->devmod[i]) {
            printf("Model: %x Option: %x\n", DQRdCfg->devmod[i],
DQRdCfg->option[i]);
        } else {
            break;
        }
    }
```

2. Create and initialize host and IOM sides

```
    DqDmapCreate(pDqe, hd0, &pBcb, UPDATE_PERIOD, &dmapin,
&dmapout);
```

3. Add channels into DMap

```
    DqDmapSetEntry(pBcb, DEVNIN, DQ_SS0IN, 0, DQ_ACB_DATA_RAW, 1,
&ioffset);
    DqDmapSetEntry(pBcb, DEVNOUT, DQ_SS0OUT, 0, DQ_ACB_DATA_RAW,
1, &ooffset));

    DqDmapInitOps(pBcb);
```

```
    DqeSetEvent(pBcb,
DQ_eDataAvailable|DQ_ePacketLost|DQ_eBufferError|DQ_ePacketOOB);
```

4. Start operation

```
    DqeEnable(TRUE, &pBcb, 1, FALSE);
```

5. Process data

```
    while (keep_looping) {

        DqeWaitForEvent(&pBcb, 1, FALSE, timeout, &eventsin);

        if (eventsin & DQ_eDataAvailable) {
            datarcv++;
            printf("\ndata %08x ", *(uint32*)ioffset);
            *(uint32*)ooffset = datarcv;
        }
    }
```

6. Stop operation

```
    DqeEnable(FALSE, &pBcb, 1, FALSE);
```

7. Clean up

```
    DqDmapDestroy(pBcb);
    DqStopDQEngine(pDqe);
    DqCloseIOM(hd0);
#ifndef _WIN32
    DqCleanUpDAQLib();
#endif
```

4 Appendix

Appendix A - Accessories

The following cables, boards, and layers are available for the DIO-40x layer.

DNA-CBL-62
2.5ft, 62-way round shielded cable; for connection to panel(s)

DNA-DIO-O22
Accessory panel for PowerDNA DIO layers; panel distributes 24 DIO channels into single group of 24 lines or, in case with 48 DIO channels, into 3 groups of 16 lines which connect to three Opto-22 compatible connectors.

DNA-STP-62
62-channel screw terminal panel

5 Glossary

A

API Application Programming Interface, a collection of high-level language function calls that provide access the functions in a driver or other utility.

B

bit One binary digit, either 0 or 1.

byte Eight related bits of data, an 8-bit binary number. Also used to denote the amount of memory required to store one byte of data.

C

coupling The manner in which a signal is connected from one location to another.

crosstalk An unwanted signal on one channel due to an input on a different channel.

current drive capability The amount of current a digital or analog output channel can source or sink while still operating within voltage range specifications.

current sourcing The ability of a DAQ card to supply current for analog or digital output signals.

I

input bias current The current that flows into the inputs of a circuit.

input impedance The measured resistance and impedance between the input terminals of a circuit.

input offset current The difference in the input bias currents of the two inputs of an instrumentation amplifier.

integral control A control action that eliminates the offset inherent in proportional control.

isolation voltage The voltage that an isolated circuit can normally withstand, usually specified from input to input and/or from any input to the amplifier output, or to the computer bus.

K

k kilo, the standard metric prefix for 1000 or 10^3, used with units of measure such as volts, Hertz, and meters.

M

M mega, the standard metric prefix for 1 million or 10^6, when used with units of measure such as volts and Hertz; the prefix for 1,048,576, or 2^{20}, when used to quantify data or computer memory.

Mbytes/s A unit for data transfer that means 1 million or 10^6 bytes/sec.

N

noise An undesirable electrical signal. Noise comes from external sources such as the AC power line, motors, generators, transformers, fluorescent lights, soldering irons, CRT displays, computers, electrical storms, welders, radio transmitters as well as internal sources such as semiconductors, resistors and capacitors.

O

optical isolation The technique of using an optoelectronic transmitter and receiver to transfer data without electrical continuity to eliminate high potential differences and transients.

output slew rate The rate of change of an analog output voltage from one level to another.

overhead The amount of computer processing resources, such as time or memory, required to accomplish a task.

P

PID control A 3-term control algorithm combining proportional, integral and derivative control actions.

PLC Programmable logic controller, a special-purpose computer used in industrial monitoring and control applications. PLCs typically have proprietary programming and networking protocols and special-purpose digital and analog I/O ports.

Polled mode DAQ card operating mode whereby the user application queries the board about the status of various subsystems as needed.

R

real time A system in which the desired action takes place immediately when all input conditions are fulfilled; it never has to wait for other processes to complete before it can start. In DAQ terms, it generally refers to the processing of data as it is acquired instead of being accumulated and getting processed at a later time.

S

SDK Software developer's kit, a collection of drivers and utilities that allow engineers to write their own application programs.

SE see single-ended.

single-ended	a term used to describe an analog-input configuration where you measure each channel with respect to a common analog ground.
S/s, S/sec	samples/sec, samples per second
system noise	A measure of the amount of noise seen by an analog circuit or an A/D when the analog inputs are grounded.
T	
TCP/IP	Transmission Control Protocol/Internet Protocol, the basic multi-layer communication protocol of the Internet but that is also used in a private network (either an intranet or an extranet). The higher layer, TCP, manages the assembling of a message or file into smaller packets that are transmitted and received by a TCP layer that reassembles the packets into the original message. IP handles the address portion of each packet so it gets to the right destination.
throughput rate	The flow of data, measured in bytes/sec, for a given continuous operation.
transfer rate	The rate, measured in bytes/sec, at which data is moved from a source to a destination after software initialization and setup operations; the maximum rate at which the hardware can operate.
U	
UCT	User counter/timer

6 Index

A

Advanced Circular Buffer 8
architecture 2

B

buffer(s) 2

D

data & word size 7

F

features 1

G

glossary of terms 13

M

mode
ACB 8
DMap 9

O

over-voltage protection 1

P

programming (low level)
commands 8
layer configuration 7

T

throughput rate 1

W

wiring 3